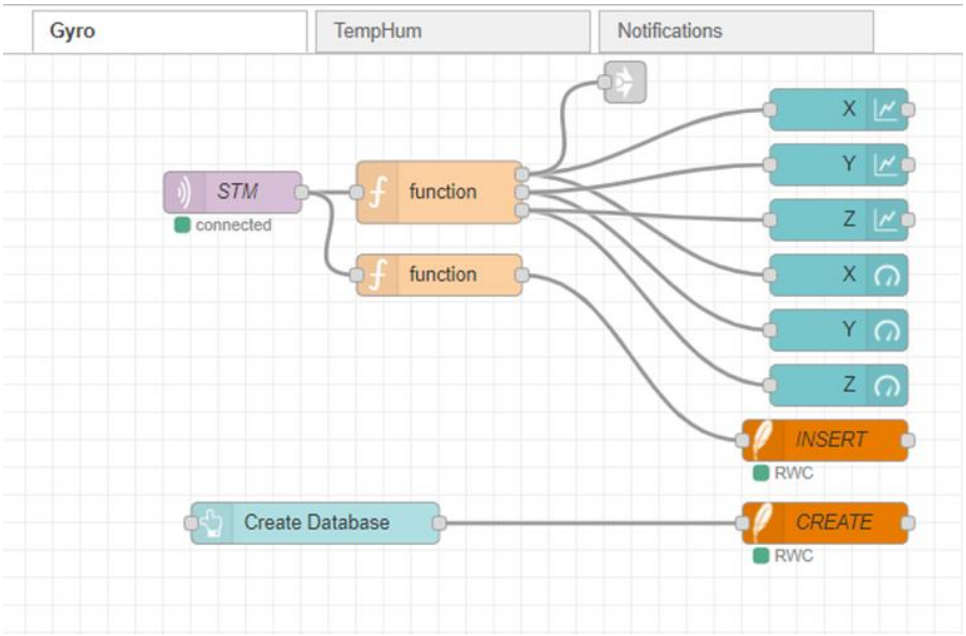
République Tunisienne
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Tunis El Manar
Ecole Nationale d'Ingénieurs de Tunis
Département Génie Electrique

Projet Réseaux Locaux Industriels

Prototype d'une interface de monitoring d'une chambre froide

Notice d'utilisation et description fonctionnelle



Mokhtar HICHRI
Firas SKHIRI
3AGE1 : Automatique et Informatique Industrielle

Année Universitaire : 2020/2021

Table des matières

Introduction:

A travers ce projet, nous allons réaliser une interface web de supervision permettant de traduire les données envoyées par la carte de développement IOT STM32L475 en utilisant le protocole MQTT.

Dépendances:

Ce programme utilise la librairie SQLITE3, qui n'est pas inclut par défaut avec Node-RED. Afin de pouvoir utiliser correctement cette interface Web, il vous faut installer les packages suivants :

- Node-RED SQLite

Commande :

```
cd ~/.node-red
npm i --unsafe-perm node-red-node-sqlite
npm rebuild
```

- DB Browser for SQLite

Afin de visualiser des bases de données SQLite, vous pouvez utiliser DB Browser for SQLite, un logiciel gratuit et open source permettant de manipuler les bases de données SQLite.

https://sqlitebrowser.org/

- Microsoft TTS Package

Si vous utilisez une machine avec Windows 10 ou Android, il vous faut un package permettant de réaliser la synthèse vocale des commandes TTS.

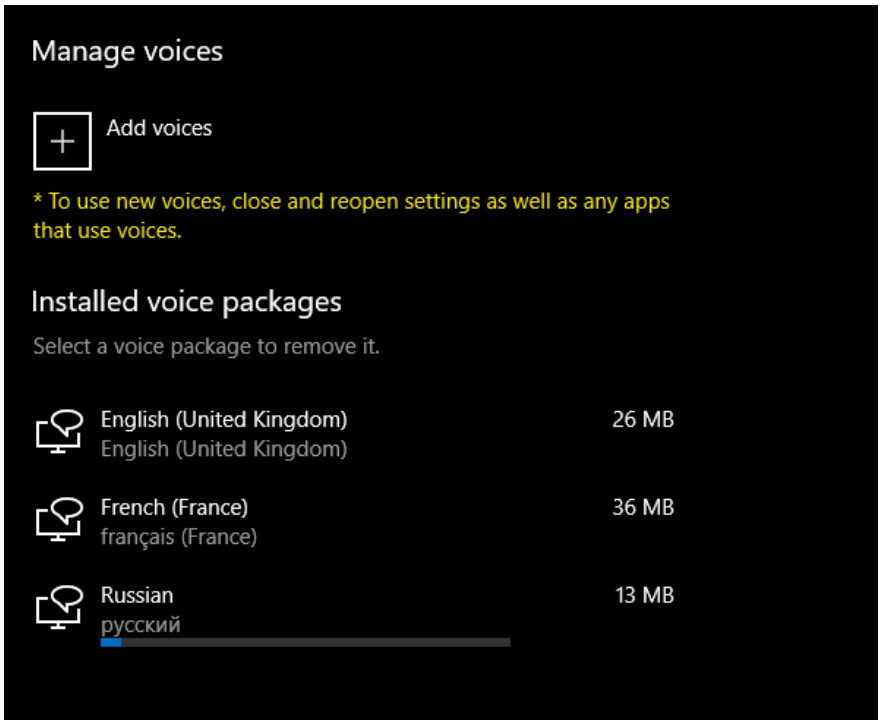
Windows 10 inclut par défaut l'option Text To Speech, de même pour Android si vous utilisez une application utilisant TTS comme Google Maps.

Si TTS n'est pas fonctionnel, prière de suivre les étapes suivantes :

Machine équipée de Windows 10 :

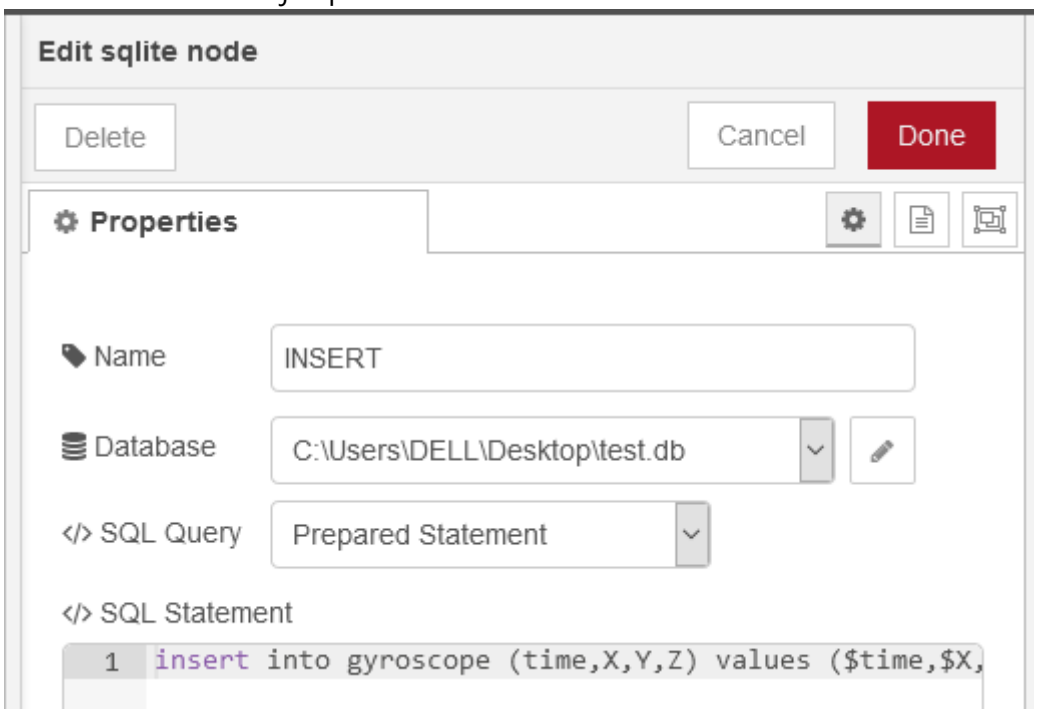
1) Naviguez au panneau de configuration.
2) Time and Language => Onglet Speech ou Voix
3) Si aucun voice package n'est installé, cliquez sur « Add voices »
4) Sélectionnez English (United Kingdom)

N.B : Si le package Français France est installé, les fonctionnalités TTS fonctionnent, mais prononçant les mots anglais avec un accent. Il est préférable d'utiliser le package English UK afin d'avoir des commandes vocales compréhensibles. La voix utilisée pour notre démonstration vidée est Microsoft Hazel.



Avant de déployer votre solution :

1) Créez un fichier .db (il suffit juste de créer un fichier .txt et de changer son extension en .db)
2) Naviguez l'interface NODE-Red jusqu'à trouver le NODE INSERT.



3) Changez l'emplacement de la base de données à celui du fichier .db que vous avez créé.
4) Faites de même pour le Node CREATE .
5) Déployez votre solution, naviguez à l'onglet diagnosis et cliquez sur le bouton « Create Database », une table prédéfinie sera ajoutée à votre fichier .db.

Votre application est désormais configurée, vous pouvez commencer à communiquer avec votre serveur MQTT.

1. Programmation de la carte de développement STM32L475 :

Afin de programmer la carte de développement Node IOT (STM32L475), nous avons utilisé un IDE avec les bibliothèques nécessaires.

Arduino IDE nous permet d'implémenter de manière simple une boucle permettant d'effectuer un envoi continu des valeurs obtenus à partir des capteurs intégrés à la carte.

Librairies utilisées :

- Inventek_Wifi
- MQTT 2.2.2
- BSP_STM32L75_IOT

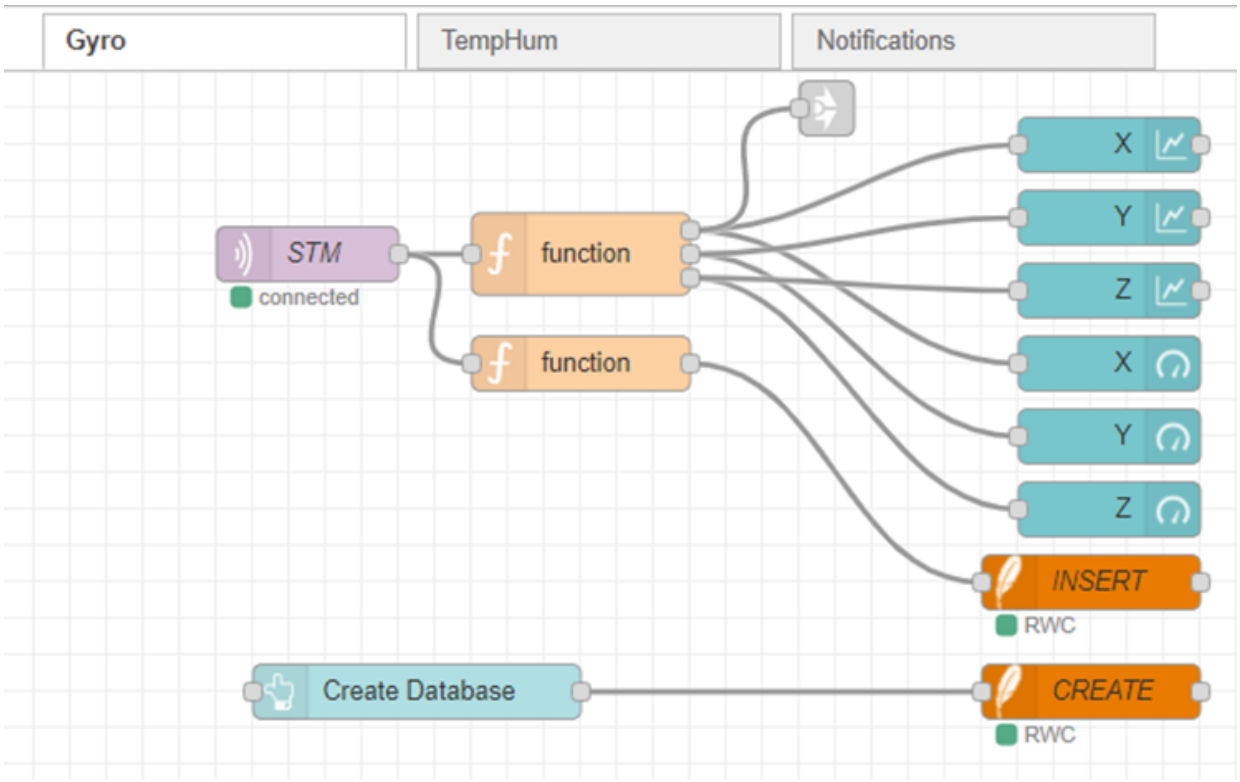
Le code que vous trouverez ne publie que les données utilisées dans cette solution. Notamment les valeurs du Gyroscope, les valeurs de température et de l'humidité.

2. Programmation Node-RED

2.1 Backend :

Etant donné que nous utilisons le protocole MQTT afin de pouvoir communiquer avec la carte STM32L475VG, nous utilisons un Subscriber au topics « /gyro », « /temperature « et « /humidity ».

Pour la clarté du code, nous utilisons deux pages :



La première page, effectue la conversion des données envoyées ainsi que l'insertion de ces données avec un timestamp dans une base de données SQL en utilisant le Node INSERT.

Le code source présent dans le Node Function précédant le Node INSERT est le suivant :

```
var payload=msg.payload;
msg.params = { $time:Date.now(), $X:payload.split(',')[0].split('(')[1],$Y:payload.split(
,')[1],$Z:payload.split(',')[2].split(')')[0]};
return msg;
```

Ensuite, le message envoyé est inséré à la table en utilisant la commande SQL suivante:

```
insert into gyroscope (time,X,Y,Z) values ($time,$X,$Y,$Z)
```

Un autre bouton permet de créer une table SQL dans un fichier .db dont le chemin est spécifié dans le Node CREATE.

Nous obtenons ainsi la table suivante :

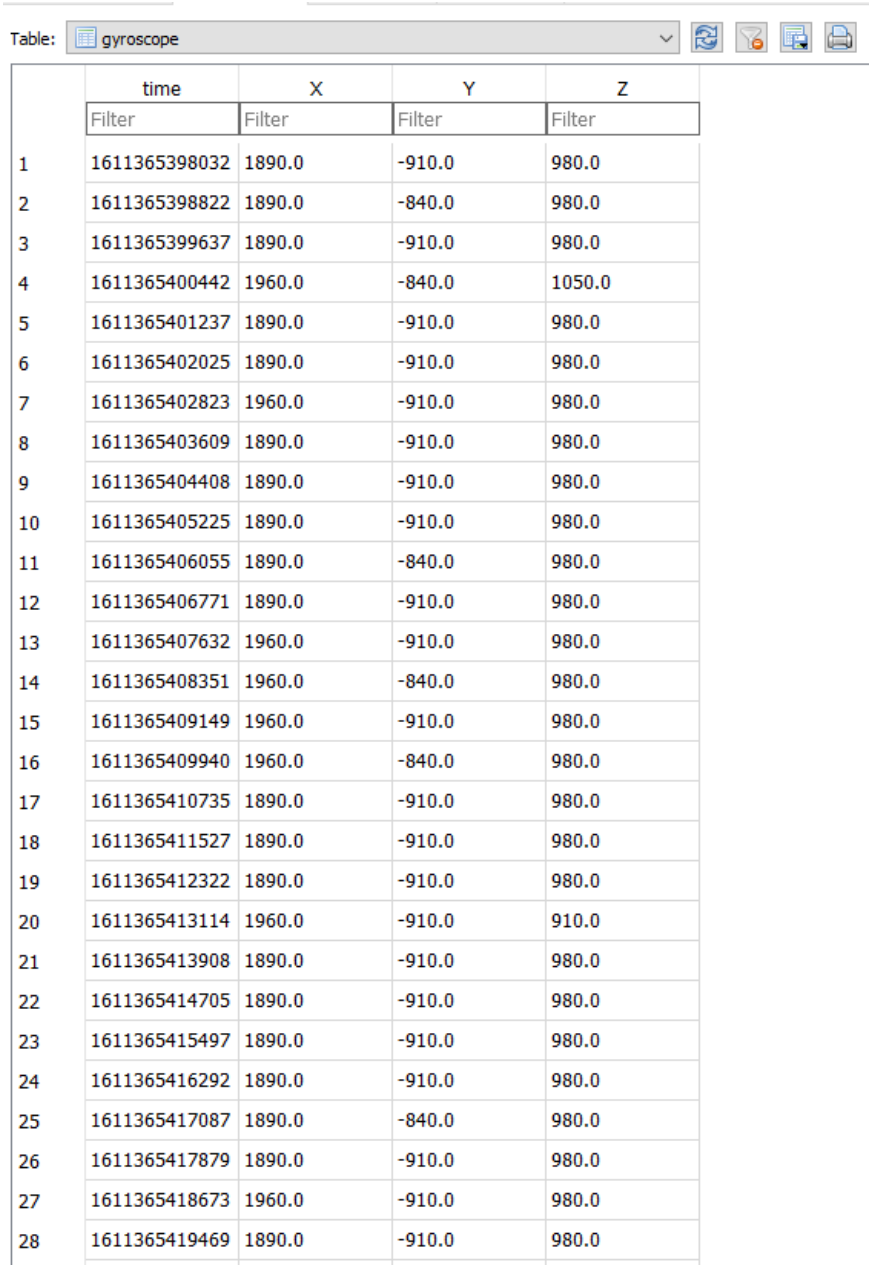
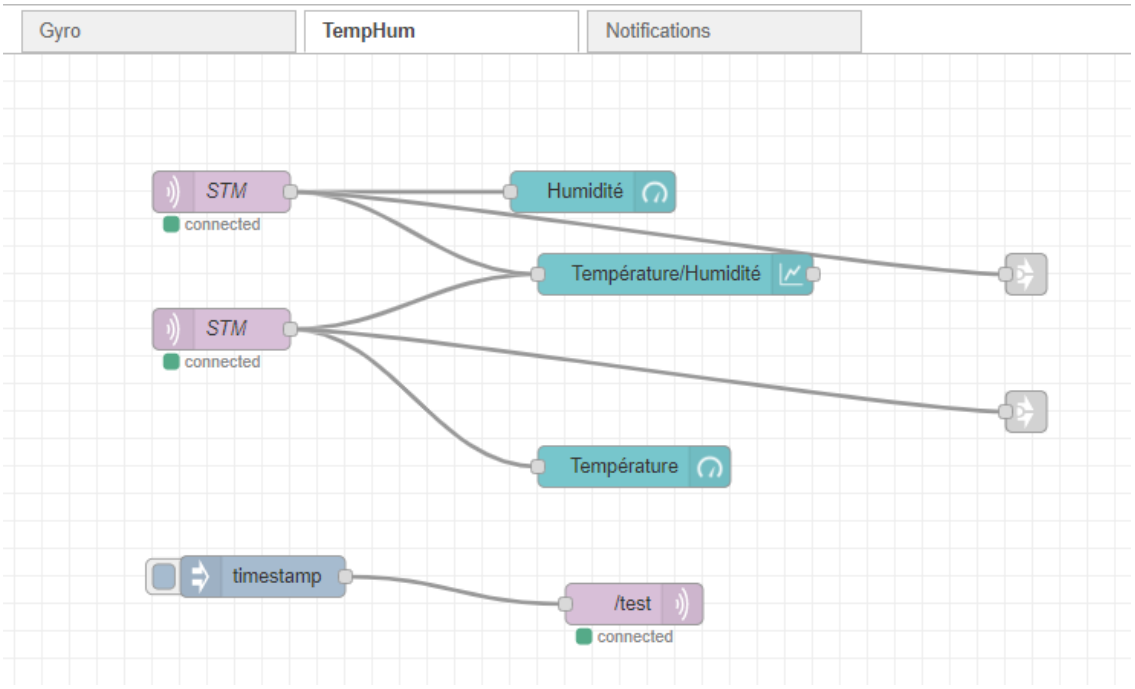
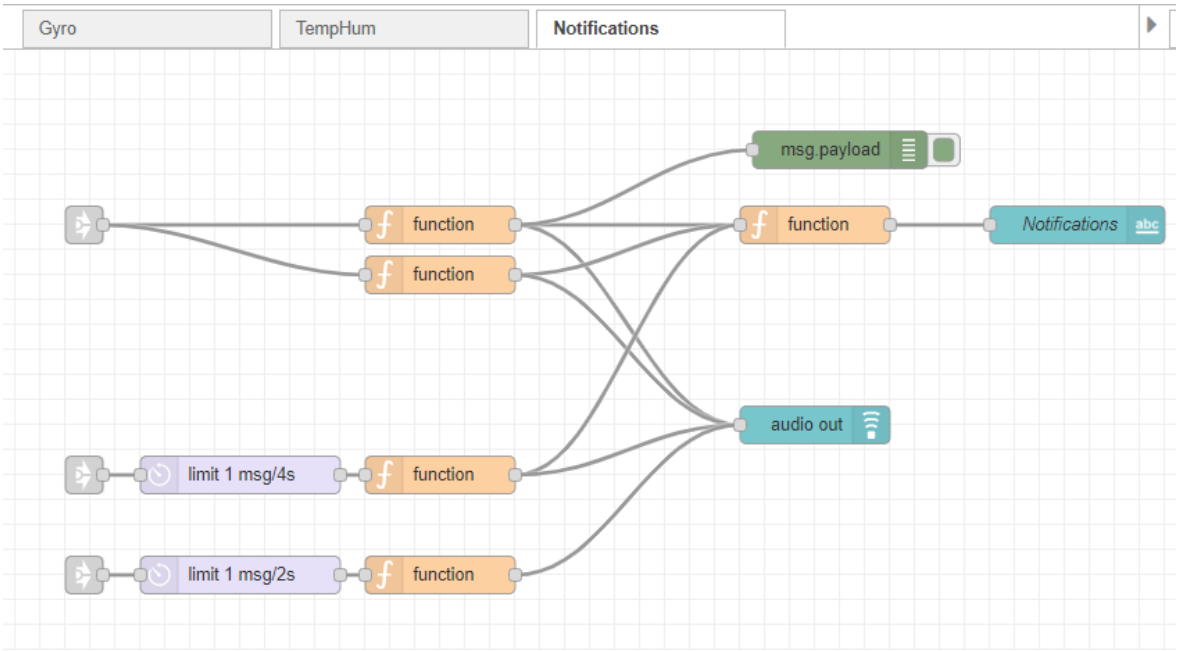


Table: gyroscope

	time	X	Y	Z
	Filter	Filter	Filter	Filter
1	1611365398032	1890.0	-910.0	980.0
2	1611365398822	1890.0	-840.0	980.0
3	1611365399637	1890.0	-910.0	980.0
4	1611365400442	1960.0	-840.0	1050.0
5	1611365401237	1890.0	-910.0	980.0
6	1611365402025	1890.0	-910.0	980.0
7	1611365402823	1960.0	-910.0	980.0
8	1611365403609	1890.0	-910.0	980.0
9	1611365404408	1890.0	-910.0	980.0
10	1611365405225	1890.0	-910.0	980.0
11	1611365406055	1890.0	-840.0	980.0
12	1611365406771	1890.0	-910.0	980.0
13	1611365407632	1960.0	-910.0	980.0
14	1611365408351	1960.0	-840.0	980.0
15	1611365409149	1960.0	-910.0	980.0
16	1611365409940	1960.0	-840.0	980.0
17	1611365410735	1890.0	-910.0	980.0
18	1611365411527	1890.0	-910.0	980.0
19	1611365412322	1890.0	-910.0	980.0
20	1611365413114	1960.0	-910.0	910.0
21	1611365413908	1890.0	-910.0	980.0
22	1611365414705	1890.0	-910.0	980.0
23	1611365415497	1890.0	-910.0	980.0
24	1611365416292	1890.0	-910.0	980.0
25	1611365417087	1890.0	-840.0	980.0
26	1611365417879	1890.0	-910.0	980.0
27	1611365418673	1960.0	-910.0	980.0
28	1611365419469	1890.0	-910.0	980.0



Une autre page permet de récolter les données Humidité et Température et de les communiquer au front end.



La dernière page gère les notifications, elle permet de générer une alerte sonore et écrite à l'utilisateur. Permettant ainsi de générer un message dont le Payload est lue en utilisant le package TTS (Text To Speech) installé dans chaque machine.

```
var doorstatus = global.get('doorstatus');
pay=msg.payload;
prev=doorstatus.payload;
if ((Number(msg.payload)>5000)&&(prev!="Closed"))
{
    msg.payload= "Door has been closed"
    doorstatus.payload="Closed";
    global.set('doorstatus',doorstatus);
    return msg;
}
```

Le code ci-dessus permet de générer les messages d'alertes en cas de déplacement de la porte. Le sens de déplacement est déterminé à partir des seuils déterminés arbitrairement (5000 pour la fermeture, -2500 pour l'ouverture).

```
var msgarray = global.get('msgarray');
pay=msg.payload;
prev=msgarray.payload;
msgarray.payload=" \<br>"+Date(Date.now)+pay+" \<br>"+prev+" \<br>";
global.set('msgarray',msgarray);
return msgarray;
```

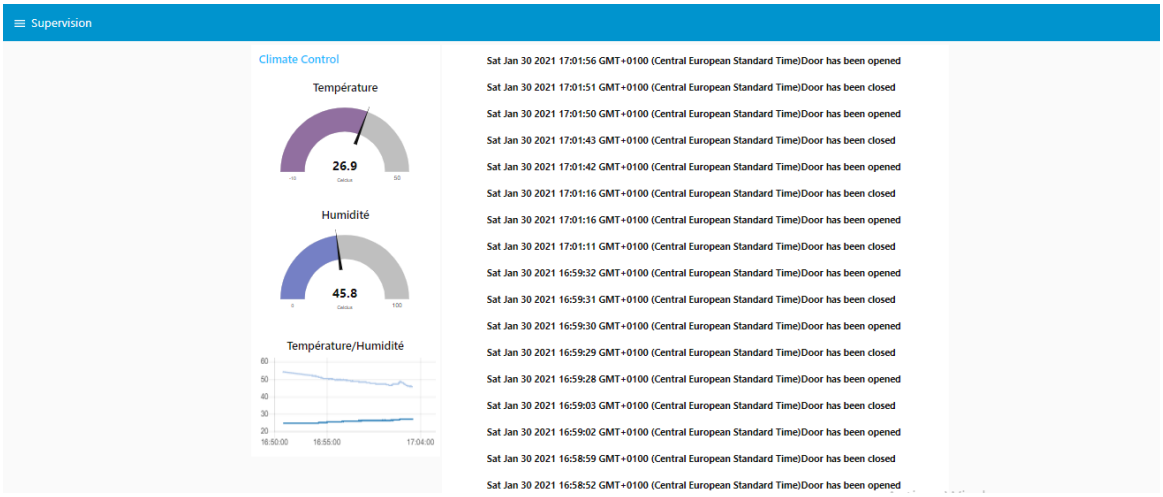
Le code ci-dessus permet d'avoir une liste à laquelle on ajoute la dernière notification avec son timestamp. Le volet notifications est décrit dans le chapitre suivant.

```
if (Number(msg.payload)>30)
{
    number=msg.payload;
    msg.payload= "Warning ! Temperature level critical ! Temperature reached is "+num-
ber +" degrees !";
    return msg;
}
```

Le code ci-dessus permet de générer des messages d'alertes envoyées au TTS ainsi qu'aux notifications en cas de dépassement de la température critique, une implémentation similaire existe pour l'humidité.

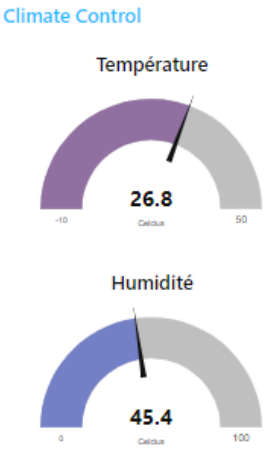
2.2 Frontend :

2.2.1 Onglet Supervision :



L'onglet Supervision contient les groupes suivants :

- **Climate Control :**



Le groupe Climate Control permet de visualiser en temps réel la valeur actuelle de la température ainsi qu'une courbe permettant d'évaluer l'évaluation des deux variables supervisées.

L'utilisation de deux jauges à gradient de couleur permettent d'avoir une meilleure interprétation des valeurs envoyées.

Les deux courbes du graphes « Température/Humidité » permettent de visualiser les valeurs pendant les 15 minutes précédentes.

- **Notifications :**

Le système d'alertes programmé en back-end permet de générer des alertes sonores et écrites. Les messages sont enregistrés dans un journal. Chaque nouveau message est placé en tête, accompagné d'un timestamp permettant de connaître l'heure de la notification.

Sat Jan 30 2021 17:01:56 GMT+0100 (Central European Standard Time)Door has been opened
Sat Jan 30 2021 17:01:51 GMT+0100 (Central European Standard Time)Door has been closed
Sat Jan 30 2021 17:01:50 GMT+0100 (Central European Standard Time)Door has been opened
Sat Jan 30 2021 17:01:43 GMT+0100 (Central European Standard Time)Door has been closed
Sat Jan 30 2021 17:01:42 GMT+0100 (Central European Standard Time)Door has been opened
Sat Jan 30 2021 17:01:16 GMT+0100 (Central European Standard Time)Door has been closed
Sat Jan 30 2021 17:01:16 GMT+0100 (Central European Standard Time)Door has been opened
Sat Jan 30 2021 17:01:11 GMT+0100 (Central European Standard Time)Door has been closed
Sat Jan 30 2021 16:59:32 GMT+0100 (Central European Standard Time)Door has been opened
Sat Jan 30 2021 16:59:31 GMT+0100 (Central European Standard Time)Door has been closed
Sat Jan 30 2021 16:59:30 GMT+0100 (Central European Standard Time)Door has been opened
Sat Jan 30 2021 16:59:29 GMT+0100 (Central European Standard Time)Door has been closed
Sat Jan 30 2021 16:59:28 GMT+0100 (Central European Standard Time)Door has been opened
Sat Jan 30 2021 16:59:03 GMT+0100 (Central European Standard Time)Door has been closed
Sat Jan 30 2021 16:59:02 GMT+0100 (Central European Standard Time)Door has been opened
Sat Jan 30 2021 16:58:59 GMT+0100 (Central European Standard Time)Door has been closed

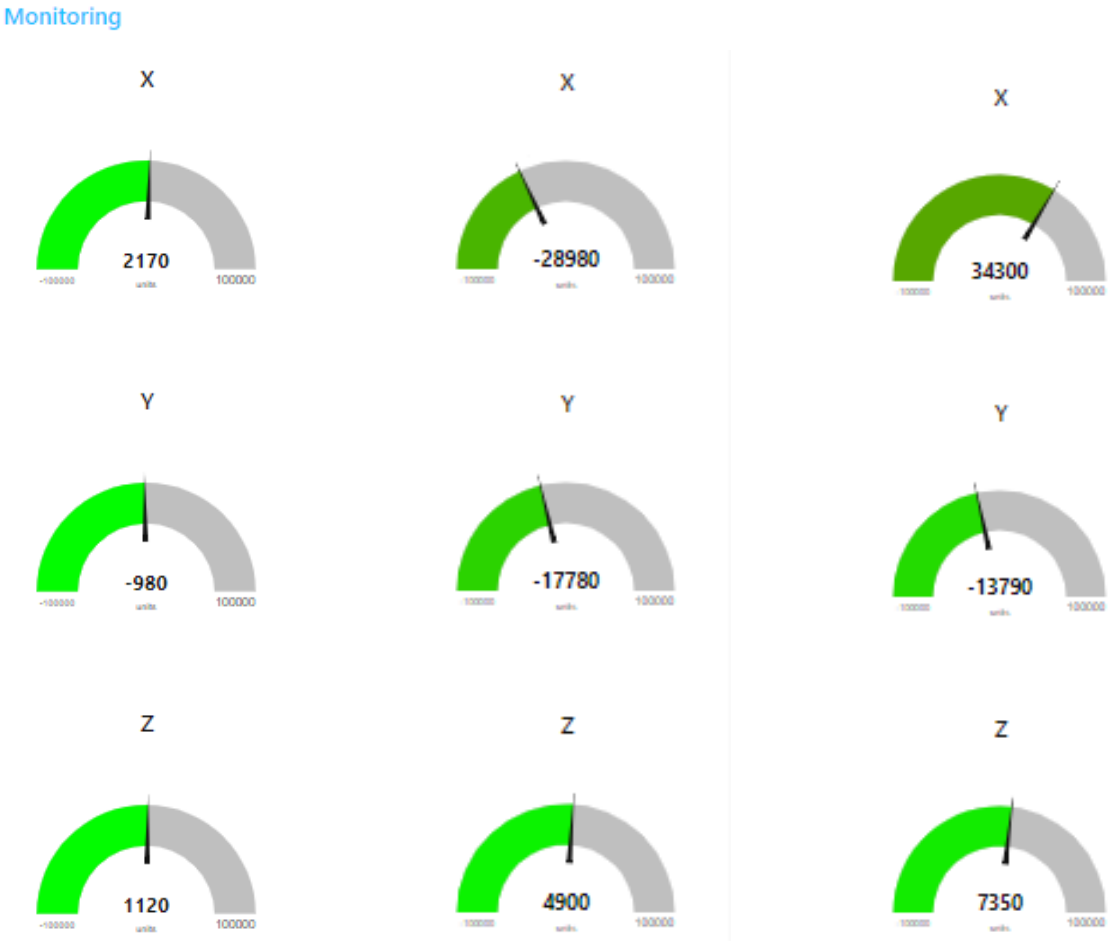
2.2.2 Onglet Supervision :

L'onglet de supervision vise à permettre à l'utilisateur d'effectuer des tests afin de pouvoir dépister les éventuels bugs et autres problèmes qui peuvent se manifester (Déconnexion de la carte, perte de connexion au réseau, erreurs d'envoi, erreurs d'interprétation, etc…). Cela en affichant les données qui seront autrement encombrantes à l'interface principale.

L'onglet Supervision contient les groupes suivants :

- **Groupe Monitoring :**

Le groupe Monitoring permet d'afficher en temps réel les valeurs des composantes X, Y et Z envoyées par la carte de développement. L'évolution de ces composantes se traduit par un déplacement de jauges colorés par un gradient. Ainsi, lorsqu'une défaillance est détectée, si l'on surveille l'évolution des jauges, nous pouvons avoir effectuer un simple diagnostic de connaître si la carte communique encore sur le réseau ou non. Nous pouvons ainsi cerner la cause du problème.



Evolution de l'état des jauges pour différentes positions de la carte

- X, Y, Z :

Un ensemble de trois graphes permettant de visualiser l'évolution des trois composantes envoyées par le gyroscope au cours du temps et cela pour les 15 dernières minutes.